SRAM SYSTEM DESIGN FOR MEMORY BASED COMPUTING

A Thesis
Presented to
The Academic Faculty

by

Muneeb Zia

In Partial Fulfillment
of the Requirements for the Degree
Masters in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
May 2013

SRAM SYSTEM DESIGN FOR MEMORY BASED COMPUTING

Approved by:

Dr. Saibal Mukhopadhyay, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Sudhakar Yalamanchili
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Hyesoon Kim
School of Computer Science
Georgia Institute of Technology

Date Approved: April 01, 2013

To my parents for their selfless love and support

ACKNOWLEDGEMENTS

I would like to express deepest gratitude to my advisor Dr. Saibal Mukhopadhyay for his constant support and guidance throughout the course of the thesis. I would also like to recognize the hard work put in by Dr. Subho Chatterjee during the design process of the chip. Furthermore, I would like to thank my friends, GREEN Lab colleagues and my family members for their constant support throughout.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

MBC	Memory Based Computing
LUT	Look Up Table
SPI	Serial Peripheral Interface

SUMMARY

The objective of the research was to design and test an SRAM system which can meet the performance criteria for Memory Based Computing (MBC). This form of computing consists of a Look-Up Table (LUT) which is basically memory array mapped with a function; the computations thereafter consist of essentially read operations. An MBC framework requires very fast and low power read operations. Moreover, the cells need to be read stable as major part of the computation is done by reading the LUTs mapped in the SRAM array.

Design and measurement of a prototype MBC test-chip with SRAM system optimized for read-heavy applications is presented in this thesis. For this purpose, a prototype MBC system was designed and taped out. Essential study of the write-ability of the core LUT is also presented. The core memory array for function table mapping was characterized for leakage, write-ability and power saving associated with pulsed read mode.

CHAPTER 1

INTRODUCTION

The choice of computing platforms, over the years, has been subject to the application. With merits and demerits associated with virtually every platform, improvement opportunities are available in all types of computing platforms.

1.1 Spatial versus Temporal Computing

Spatial computing refers to the platform where various computational blocks are spatial separate and collaborate to determine the final computation. The re-configurability comes from the software controlled interconnect network that determines the connections between fixed logic blocks. The main advantage of spatial platform is that it can take advantage of the parallelism and can compute multiple operations simultaneously. However, due to the heavy dependence on interconnects, the performance of the spatial computing platform is inherently dependent on the interconnect delay. This becomes more significant for lower technology nodes as the interconnect delay does not scale in the same manner as the delay for logic blocks. On the other hand, temporal computing uses multiple clock cycles to perform a computation in fewer spatially separated blocks. This gives the advantage of avoiding the excessive interconnect delay which otherwise would limit the maximum achievable performance. Fig. 1 shows the basic structure of the two different modes of operation for the platforms for the computation of the same function $Ax^2 + Bx + C$.

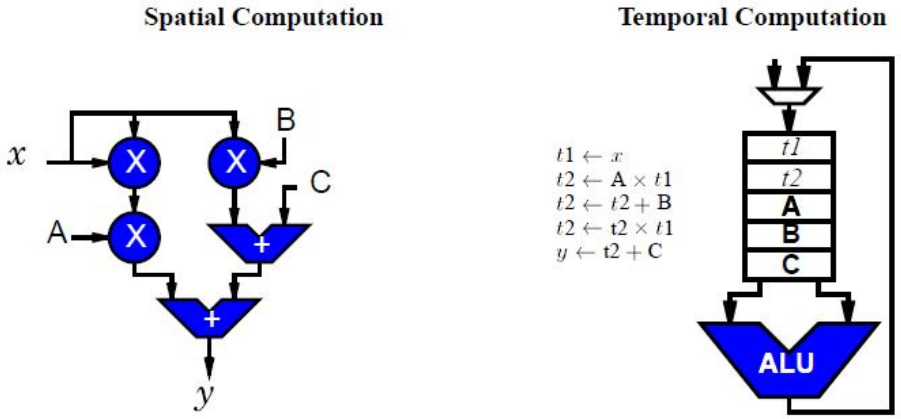


Fig. 1 Spatial versus Temporal computation for the expression $Ax^2 + Bx + C$ [1].

As depicted by the figure, the spatial computation platform uses three multipliers and two adders to compute the expression. These logic blocks are physically separated in space and require inter-block communication to compute the final result. On the other hand, the temporal platform uses one spatial block but instead computes the expression over multiple cycles. This obviously requires the need to store intermediate data if it is required for any future computation. This need will also be highlighted when the prototype MBC system is discussed in chapters 3 and 4.

1.2 Reconfigurable Computing

Traditionally, computations have been implemented in hardware (e.g. custom VLSI, ASICs, gate-arrays) or in software running on processors (e.g. DSPs, microcontrollers, embedded or general-purpose microprocessors). More recently, however, Field-Programmable Gate Arrays (FPGAs) introduced a new alternative which mixes and matches properties of the traditional hardware and software alternatives. Using FPGAs for computing led the way to a general class of computer organizations which we now call reconfigurable computing architectures. These encompass both spatial and

temporal computing platforms in the broader picture. The key characteristics distinguishing these machines are:

- The computation fabric – can be changed after fabrication
- The fabric – utilize a large degree of spatially customized computation

This class of architectures is important because it allows the computational capacity of the machine to be highly customized to the needs of an application. As single-chip silicon die capacity grows, this class of architectures becomes increasingly viable, since more tasks can be implemented spatially [1].

As mentioned, Field Programmable Gate Arrays (FPGAs) have become the most popular reconfigurable computing platform. However, the downside of the reconfigurable nature is that the design mapped on a FPGA platform operates roughly 3 times slower, occupies 10 times more area and consumes almost 2 times the power compared to an ASIC implementation at the same technology [2]. The primary reason behind such a penalty is the programmable interconnect network connecting multiple CLBs. As the process shrinks, this interconnect delay does not scale in the same manner as the logic delay. Therefore the contribution of the interconnect delay is expected to increase in future generation of FPGAs fabricated in nanometer technologies [2-3].

Memory Based Computing is a viable alternative to address the problems associated with the other reconfigurable platforms discussed above.

1.3 Reconfigurable Memory Based Computing

Memory based computing refers to the branch of computing where the function is stored in a dense memory array as a Look-up Table (LUT) and computation is done by

simple memory reads. Fig. 2 shows a very simplistic structure of a typical Memory Based Computing system.

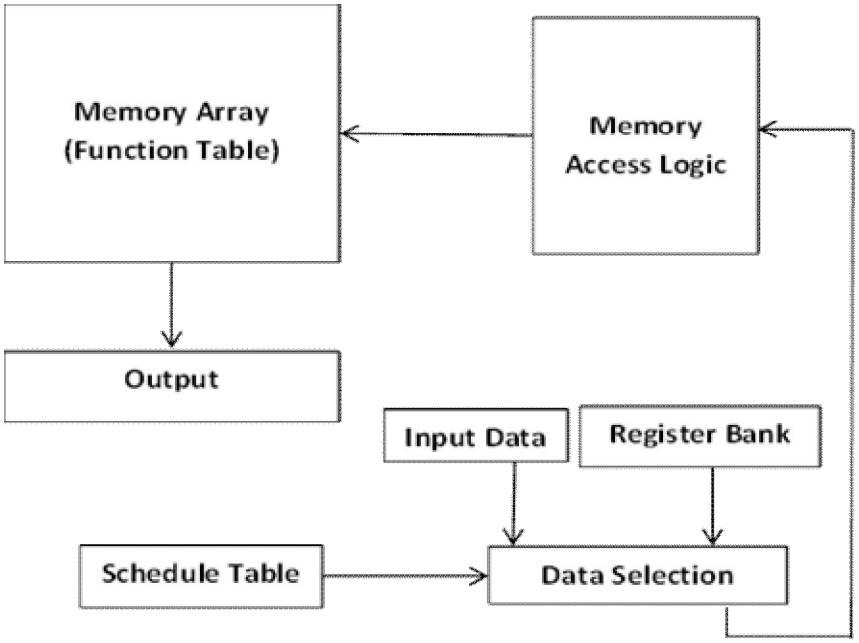


Fig. 2 Simplistic structure of a typical MBC platform

The major advantage here is that the overhead in the form of interconnect delay can be avoided if computation is done in a temporal fashion. Specifically, this can be extremely beneficial for more complex functions e.g. trigonometric, square root etc. as these functions have fewer inputs and can be easily implemented using a LUT instead of a logic block. Moreover, as most data has temporal locality, this can be exploited to reduce computational overhead. The fact that memory serves as the heart of computation, very little changes in the hardware design is required for implementation. [4]. A detailed analysis of the performance and energy improvement gained using MBC can be found in [4].

CHAPTER 2

PRIOR WORK

Memory Based Computing has been evolving as a viable alternative to the traditional computing platforms. Researches have been done on architectural, circuit and software level to make MBC substantially viable compared to the other platforms.

2.1 Architecture Level Work

The methodology used for computation in MBC can greatly affect the efficiency of the system. In [7], a content addressable memory framework was presented that made a tradeoff between purely spatial and temporal frameworks. The proposed scheme significantly reduced the memory requirement while compromising only slightly on the delay. Similarly, MBC as an accelerator in multi core architecture was presented in [4]. In [6] Somnath Paul and Swarup Bhunia presented MBARC, a reconfigurable memory-based computing model for emerging nanoscale devices, which are amenable for memory design. The proposed model provides dynamic re-configurability and minimizes the requirement for programmable interconnects as well as interfacing hardware [6]. They also discussed the use of multiple memory banks and time multiplexed execution within MCB to get an overall higher throughput.

2.2 Circuit Level Work

Circuit level modifications, especially for the LUT, have been proposed to make the core function table more stable and viable for MBC applications. A novel SRAM cell was proposed in [5] to take advantage of the read dominant nature of MBC applications.

This cell is shown in Fig. 3 which takes advantage of the fact that in a typical MBC framework, there are significantly more reads than writes. It has decoupled read and write paths which allows greater read stability. When the cell needs to be read, RWL is made low and based on whether read node is storing ‘0’ or ‘1’, the RBL will either stay high or discharges respectively. The node connected to the gate of M5 now sees the much higher gate impedance and less leakage compared to a regular 6T cell and hence is more stable. During writing, WWL is made high WBL is made ‘0’ or ‘1’ depending on what needs to be written. The write here is single ended and hence more difficult than standard 6T cell.

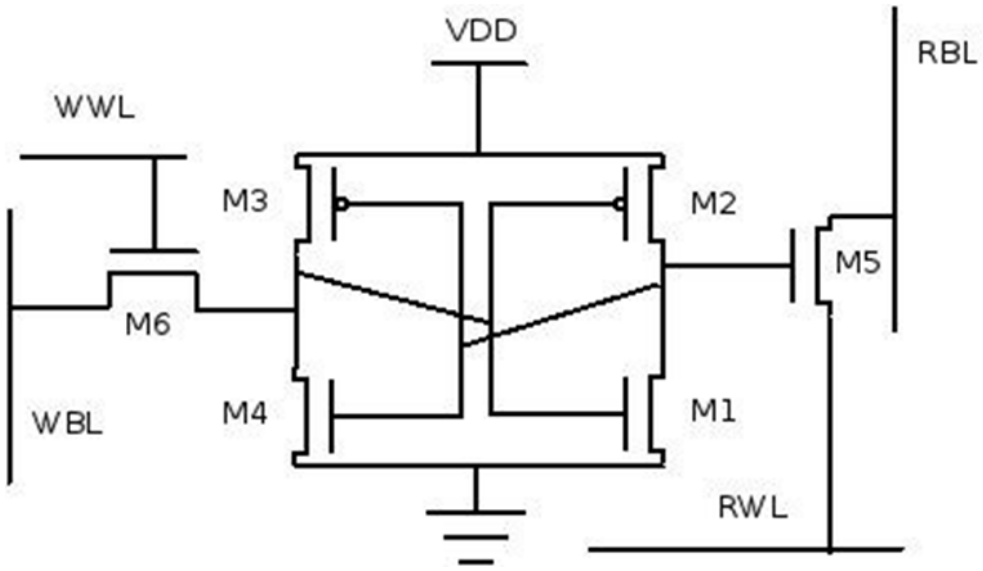


Fig. 3 Asymmetric SRAM cell for read dominant LUT application

2.2.1 The Problem of Read Sneak Path

In addition to the difficult write, this particular configuration presents an additional challenge. The read operation for this cell also presents a challenge. As seen from Fig. 4, if multiple cells connected to the BL are storing ‘1’, the BL will be charged

by these cells if the voltage drops below V_{dd}-V_{th}. This essentially means that the read operation for the system should occur before the BL falls to this value and a 'sneak' path is developed. This read sneak path can not only increase power dissipation, but can also result in functionality failure.

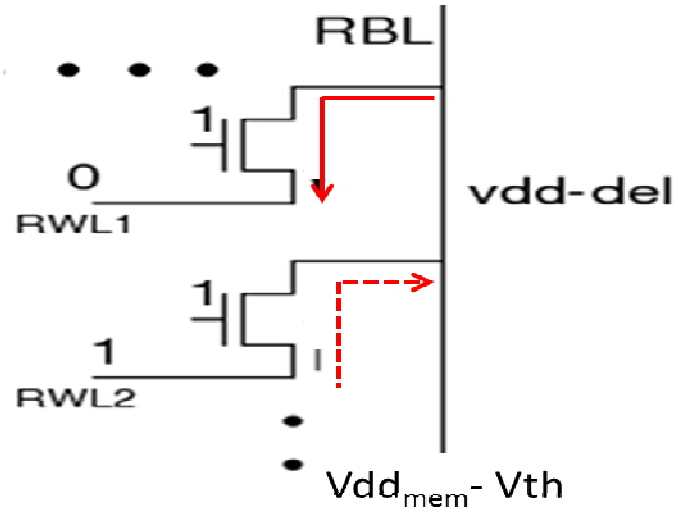


Fig. 4 Read sneak path issue associated with the used cell [5].

2.2.2 Pulsed Read Operation to Avoid Read Sneak Path

The read sneak path can be avoided by using a pulsed read operation as suggested in [5]. In the discussed MBC system, we used a delay chain to produce a pulse. The delay chain was supplied by a separate V_{dd} with which the pulse width of the pulse could be controlled. This gave us the ability to modulate the pulse width of find the "sweet spot" for optimum read operations. As shown in Fig. 5, there is an optimum pulse width which will result in least power dissipation while ensuring correct operation.

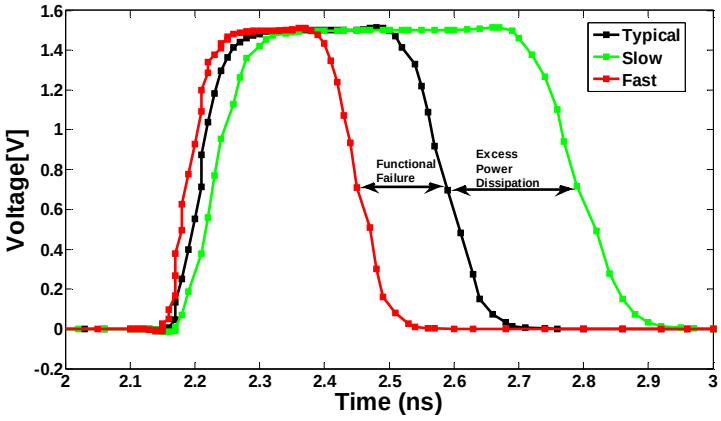


Fig. 5 Optimum pulse width for correct read operation

For the worst case where a single cell is discharging, a pulse width of 400ps was required. The pulse generation circuit is shown in the Fig. 6.

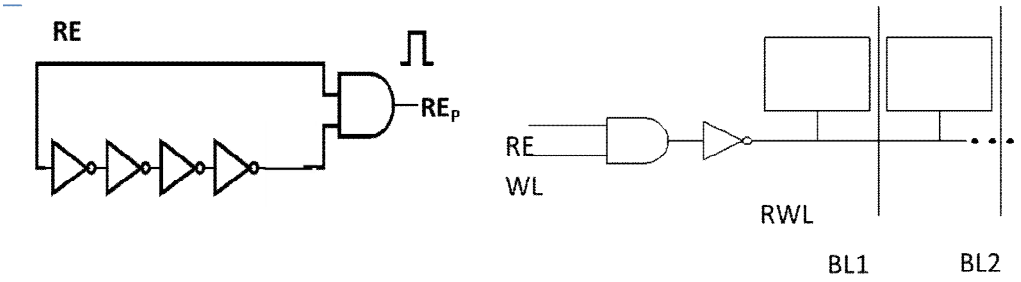


Fig. 6 Pulse generation circuit used for the MBC system

As discussed earlier, the pulsed read generation circuit consists of the clock delayed with respect to itself. The width of the pulse is controlled by an externally supplied voltage which essentially controls the delay of the chain. From the simulations we find, a delay chain element voltage supply of 0.8 V is required to produce a 500ps pulse. The pulse and the reading scheme is illustrated in Fig. 7. As seen from the figure, a read pulse is applied for a fraction of the clock cycle which prevents the bitline to discharge completely hence preventing the read sneak path problem.

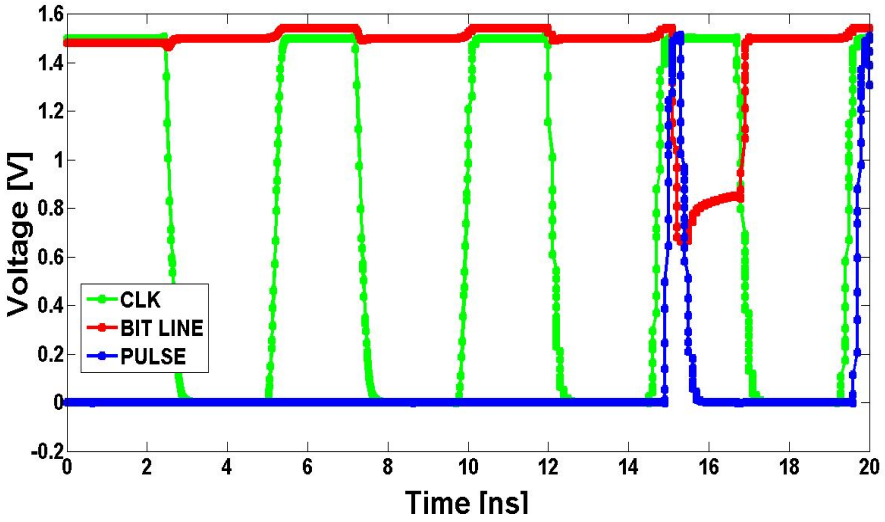


Fig. 7 Pulsed reading scheme for the MBC test chip

2.3 Software Level Work

Various algorithms have been proposed on the software side to make the overall system efficient. A heuristic algorithm for content aware mapping was proposed in [5] to exploit the fact that for the proposed LUT, the read performance was better when storing a '0' rather than a '1'. It was also discussed in [5] that energy efficiency in conventional FPGA frameworks is primarily limited by the contribution from the programmable interconnects, which suffer from poor technological scalability in terms of power and performance.

2.4 Contributions of this Thesis

This thesis builds on the circuit and architectural aspects for MBC platform. Design and measurement of a prototype MBC test-chip with SRAM system optimized for read-heavy applications is presented in this thesis. For this purpose, a prototype MBC system was designed and taped out. Essential study of the write-ability of the core LUT is also presented. The core memory array for function table mapping was characterized for

leakage, write-ability and power saving associated with pulsed read mode. The possible extensions of the work are then presented in chapter 5.

CHAPTER 3

A PROTOTYPE MEMORY BASED COMPUTING TEST-CHIP

A 2kb prototype MBC block is designed using the SRAM array as the LUT. The system level diagram of the implemented framework is shown in Fig. 8 below. The core LUT SRAM array was made from the cell proposed in [5] which had decoupled read and write paths and was much more read stable than the standard 6T cell.

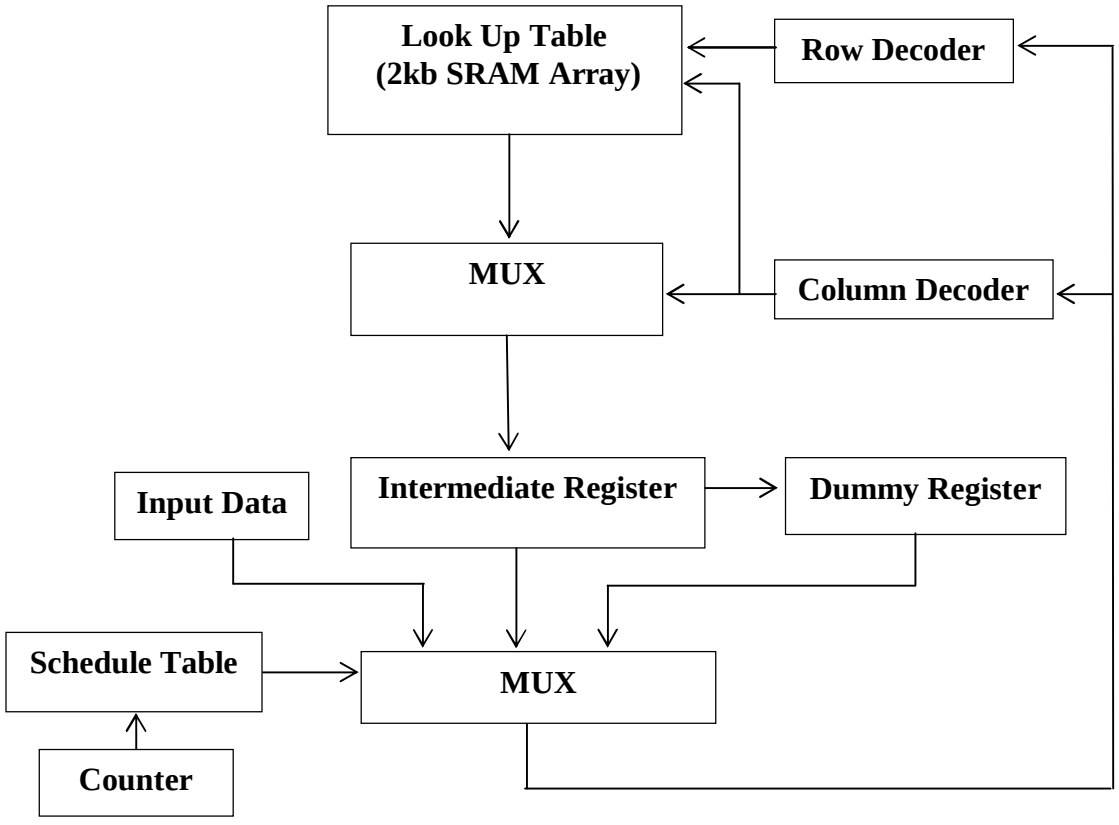


Fig. 8 System block diagram for the implemented MBC framework

3.1 System Setup

The system test setup is shown in the Fig. 9. As seen from the figure, the complete setup consisted of two serial to parallel interface blocks, which were in turn connected to PC for writing or reading.

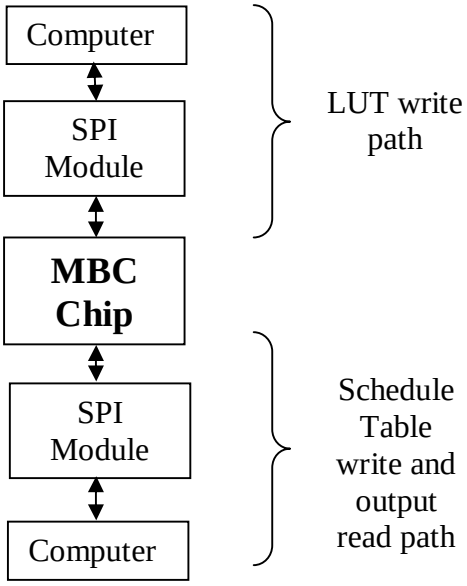


Fig. 9 System test setup block diagram

The LUT and the Schedule table were written via the Serial Peripheral Interface (SPI) module. The SPI registers on chip were interfaced with SPI (I^2C based) module for communication with the computers. Within the design, the SPI write registers for write path SPI module were hard wired to the write bit lines for the memory. Thus, whatever function was supposed to be mapped onto the LUT, the values were written to the SPI write registers and then write enable was made high. Row address was then incremented and next row written. This was repeated till the entire LUT was written. The detailed test scheme is given in the appendix.

3.1.1 Modes of operation

Table 1 shows the modes of operation for the system. The second SPI module was used to write the schedule table. Schedule table is essentially the heart of the framework. It determines which data is sent to the decoders and hence which value is evaluated by the system.

Table 1 Modes of operation for the test chip

Mode of Operation	**Remarks**
Both External Inputs	Both operands are externally provided by the user
External + Intermediate Register Inputs	One operand is external while the other operand is taken to be the value stored in the intermediate register

This second mode of operation mentioned in Table 3.1 gives the system capability to do temporal computation over multiple cycles and hence avoid communication between blocks. The function mapped onto the LUT can be used multiple times with changing inputs to compute over multiple clock cycles. When a different function has to be computed, the write SPI registers are given a new set of values and the write process is repeated for the entire LUT.

3.2 Importance of SRAM in MBC

The above discussion highlights one of the key aspects of MBC – multiple reads of LUT. The advantage of reduced interconnect delay as opposed to the spatial computing platform comes from the fact that once a LUT is written, it is read multiple

times over number of clock. As a result, most MBC applications end up having an extremely high read to write ratio.

This creates the necessity of having an SRAM system that is more stable and has a lower delay during the read operation. For this reason, cell shown in Fig. 3 was used in the design. This cell was originally proposed in [5]. The major advantage of this cell is the decoupled read and write. This inevitably makes the write process more difficult, however, due to the high read to write ratio, the overall system performance is not affected significantly. On the other hand, this decoupling makes the cell much more read stable. As seen in Fig. 3, the read node is connected to the gate of the read access transistor. Hence the leakage of the storage node is considerably less compared to the standard 6T cell. However, as the write is now single ended, it is difficult to write to the cell. This can be taken care of by boosting the word line.

3.2.1 Schedule Table – Functionality and Importance

As discussed in section 3.1.1, schedule table is the main control center for all the operations executed by the system. For our test chip, this is written by the user via the SPI modules. The sample schedule table output table is shown in Fig 10. Each output of the schedule table corresponds to a different set of control signals applied to the MUXs. As discussed in section 3.1, these operations could be between the IR content, DR content or external inputs.

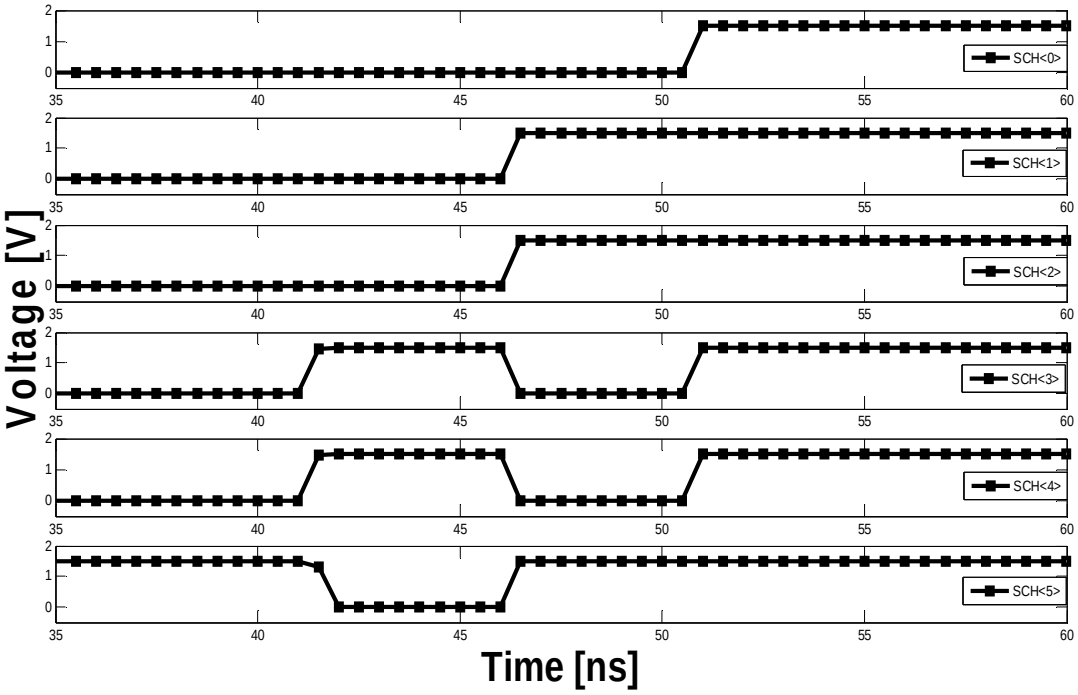


Fig. 10 Sample output for the schedule table

Fig. 11 shows a sample multi-cycle operation implemented on the MBC platform. A simple case of the addition of 0001 and 1111 and then addition of 0001 to the result is taken. At first, both IR and DR have the value 0000 stored in them. At the start of the memory read operation, IR receives the read data and stores the sum of the two inputs i.e. it stores 0001 0000.

In the next clock cycle, the value of IR is shifted to the DR. This allows use of the intermediate value for any future computation as IR is over written at every memory read operation.

In case of a repetitive addition, the value stored in the DR and one of the external inputs is selected by the schedule table and is sent to the decoders. In the current example of addition, this implies that in the second clock cycle, schedule table will select the DR value and the input 0001 to be decoded and the result from the memory stored in the IR.

As seen from the Fig. 11, at the second clock edge, the value stored in the IR is 0001 which is basically the sum of the lower nibble of DR and the input. This essentially is the result of our computation and can be read out by reading the IR and DR. Fig. 11 shows the value of IR moved to DR on the third clock cycle as well where the data in IR is overwritten again. This is presented to highlight the need of a dummy register to temporarily store values that might be needed in future computations. However, it is worth mentioning that this is not necessarily a general requirement for any system but in fact depends on the type of architecture and number of intermediate registers chosen for a certain design.

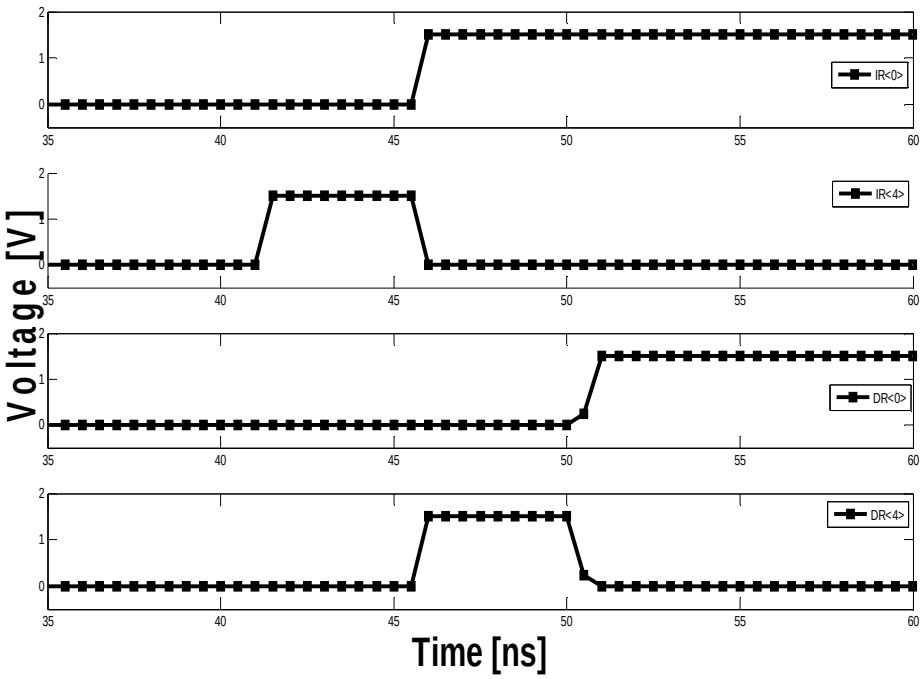


Fig. 11 Sample multi-cycle addition of 0001 and 1111

CHAPTER 4

CHIP CHARACTERIZATION

The full chip layout is shown in Fig. 12 and the packaged chip is shown in Fig. 13. Table 2 summarizes the system specifications for the MBC chip. The prototype chip was characterized for various parameters. The primary aim was to characterize the SARM array being used for the LUT. Hence data was collected for the leakage, write-ability and the pulsed read operation and is discussed below.

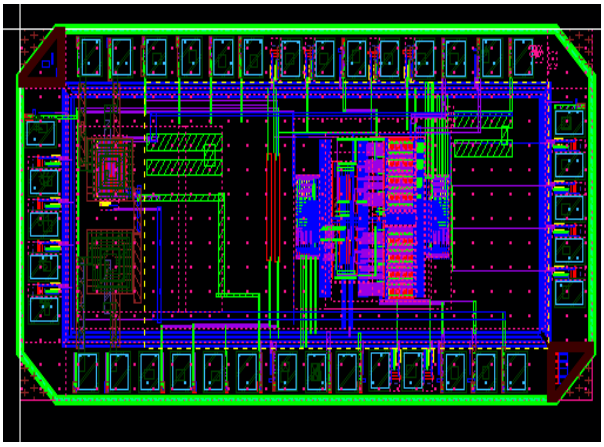


Fig. 12 Full chip layout for the MBC system

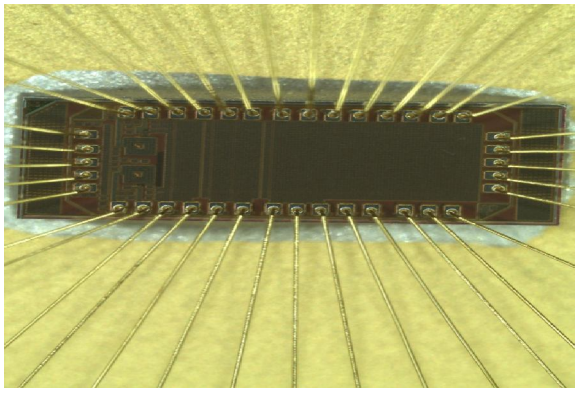


Fig. 13 Packaged view of the MBC chip

Table 2 MBC system chip specifications

Technology	IBM 130nm
Transistor Count	~35000
Chip Area	1mmX2mm
Nominal Operating Voltage	1.2V (Memory) 1.5V (Logic)
Operating Frequency	~190MHz

4.1 Cell Characterization

Fig. 14 shows the butterfly curve for the SRAM cell used in the MBC test chip. From the curves, the maximum static noise that can be sustained by the cell without flipping the data is approximately 0.45V.

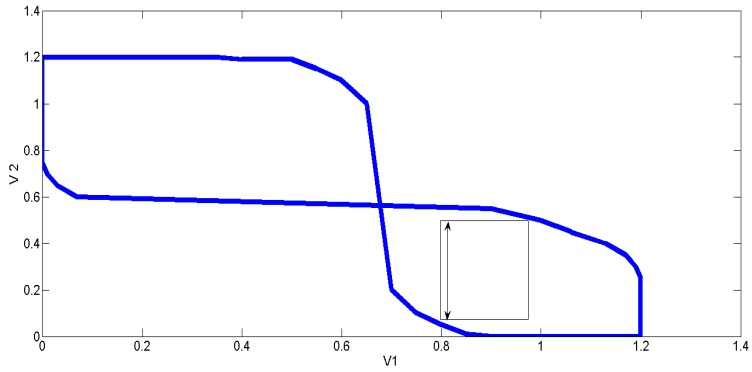


Fig.14 Butterfly curve for the SRAM cell used

4.2 System Clock

Fig. 15 shows the system clock. The clock was generated using a simple ring oscillator circuit. This circuit was provided with a separate V_{dd} to control the frequency.

Fig. 16 shows the variation of clock frequency with varying voltage. As seen from the figure, the system operates at approximately 190MHz at a nominal voltage of 1.2V.

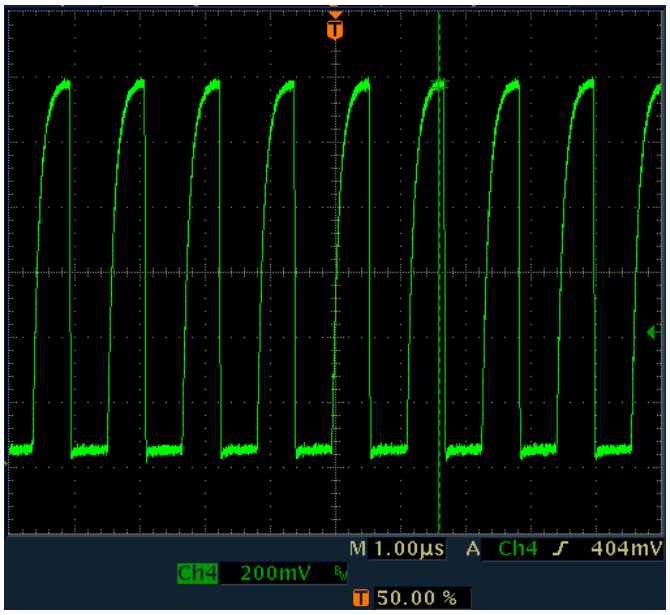


Fig. 15 System clock observed externally

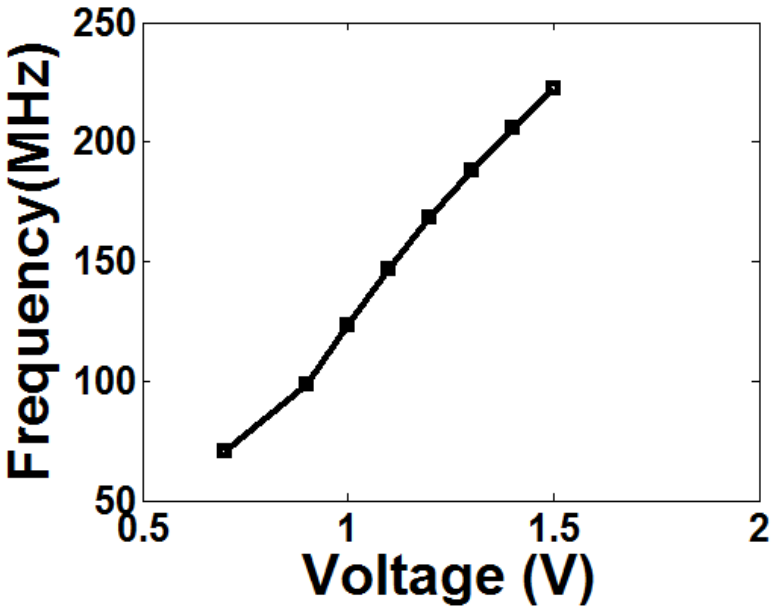


Fig. 16 Voltage versus frequency plot for the chip clock

4.3 Memory Array Leakage

Fig. 17 shows the leakage current of the memory array. The memory array was supplied with a separate V_{dd} in the design to determine the leakage of the memory array with the modified cell as the building block.

Overall, the leakage current shows a linear increase with the increased memory voltage. It is essential to note that because at a certain time, only few memory locations are being read, the leakage from the rest of the cells is a considerable fraction of the total power dissipated in the memory array. Moreover, as the function table needs to be mapped onto the memory prior to any computation and has to hold its value for all the subsequent reads, having a lower baseline leakage will help reduce the overall power dissipation.

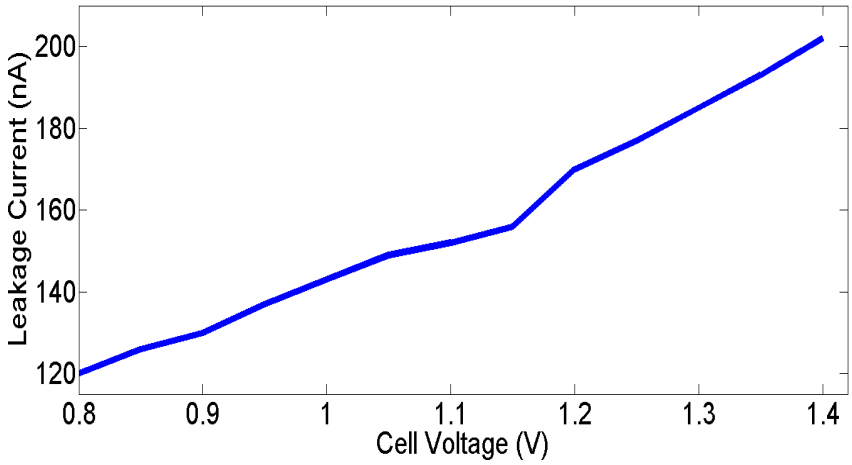


Fig. 17 Leakage current for the memory array (LUT)

4.4 Write-ability of the Memory Array

As discussed earlier, the modified SRAM cell used in the MBC implementation here has single ended read and write. This makes write much more difficult than the conventional SRAM cell and requires word line boosting to be written properly. This necessitates the study of the write-ability of the LUT across various operating points. Tests were carried out to determine the word line boosting required at a certain memory voltage for different clock frequencies. Memory voltage was fixed for a particular operating frequency and then the WL voltage required to write was recorded. The results are shown in Fig. 18. The results show that at lower clock frequencies, the LUT is written with relatively less word line boosting required. As the frequency is increased, more word line boosting is required for functionally correct write-ability.

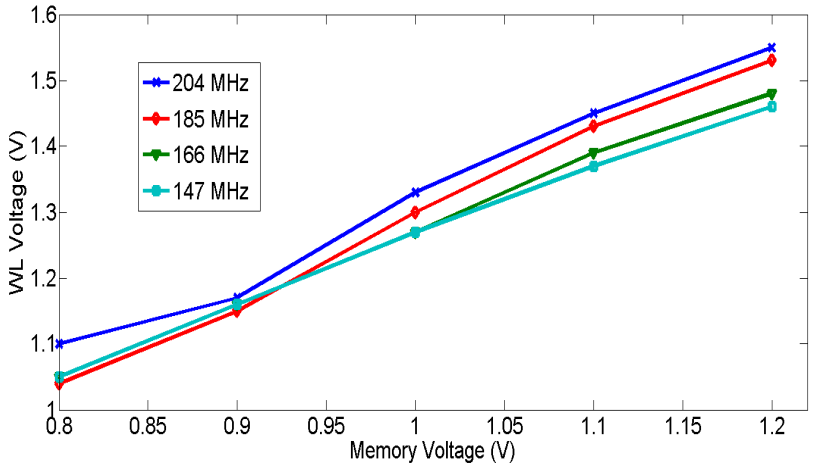


Fig. 18 Word line boosting required for various operating frequencies

4.5 Pulsed Read Power Saving

As discussed earlier, a read pulsed scheme was implied in the system to avoid read sneak path condition. The chip was tested for power dissipation with and without the read pulsed scheme. Specifically, the power associated with the logic circuitry at a nominal supply voltage of 1.2 V and an operating frequency of 190 MHz was measured for the pulsed scheme and was compared to the power dissipation without the scheme for the chip. The result of the normalized power in both cases is shown in Fig. 19. There is an overall 9% saving in read power while using the pulsed read scheme for the prototype chip. Fig. 20 also shows how the logic power dissipation varies when the V_{dd} pulse is varied. Higher voltage corresponds to a larger pulse width.

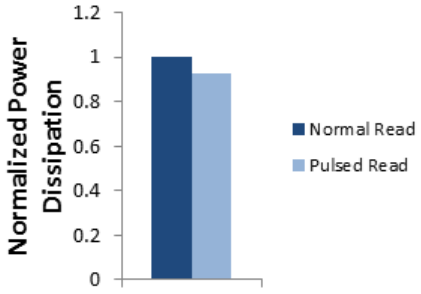


Fig. 19 Power dissipation comparison with and without pulsed read

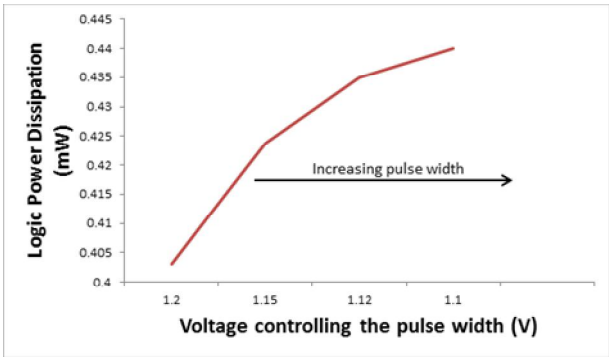


Fig. 20 V_{dd} pulse versus logic power dissipation

CHAPTER 5

CONCLUSION AND FUTURE WORK

This thesis highlighted the importance of Memory Based Computing as a viable alternative to the traditional computing platforms and pointed out various reasons for its feasibility. The core work presented here dealt with the design and characterization of a prototype chip for the demonstration of a test system. Specifically, the SRAM array for the system was characterized.

5.1 Summary of Contributions

A prototype MBC test-chip with SRAM system optimized for read heavy applications was designed, taped out and characterized for various parameters. The core memory array for function table mapping was characterized for leakage, write-ability and power saving associated with pulsed read mode.

5.2 Future Work

A basic platform has been implemented during the course of this work. However, there a number of extensions that can eventually prove essential for furthering the viability of MBC.

5.2.1 Architectural Extensions

The system designed and tested here was very basic and simplistic. The number of intermediate registers were kept to a meager 2 avoid additional complexity in the system. However, for more complex computations over far greater number of cycles, the number of intermediate storage registers can be greatly increased which will give system

the capability of performing much more complex computations. Moreover, the entire architecture developed here can only handle 4 bit external operands without losing critical information for any input. A new architecture capable of handling higher number of bits input can be designed which can make the system capable of manipulating larger data sets.

5.2.2 Circuit Level Extensions

The core LUT in the presented design was made using the alternate 6T cell structure proposed in [5]. Although the presented cell has lesser leakage compared to the standard 6T cell, there is still a significant improvement possible for leakage reduction in the core memory. As highlighted earlier, a large part of the main memory is ‘idle’ while only a few bits are being read during any computation. This consequently diverts the attention to employ suitable leakage reduction techniques for the memory. An input aware power gating mechanism can be incorporated in the system that power gates the ‘idle’ rows/columns to their Data Retention Voltage (DRV) while keeping the used rows/columns active. Moreover, a novel SRAM cell that can power gate itself based on the data stored can also be tested for feasibility in such application.

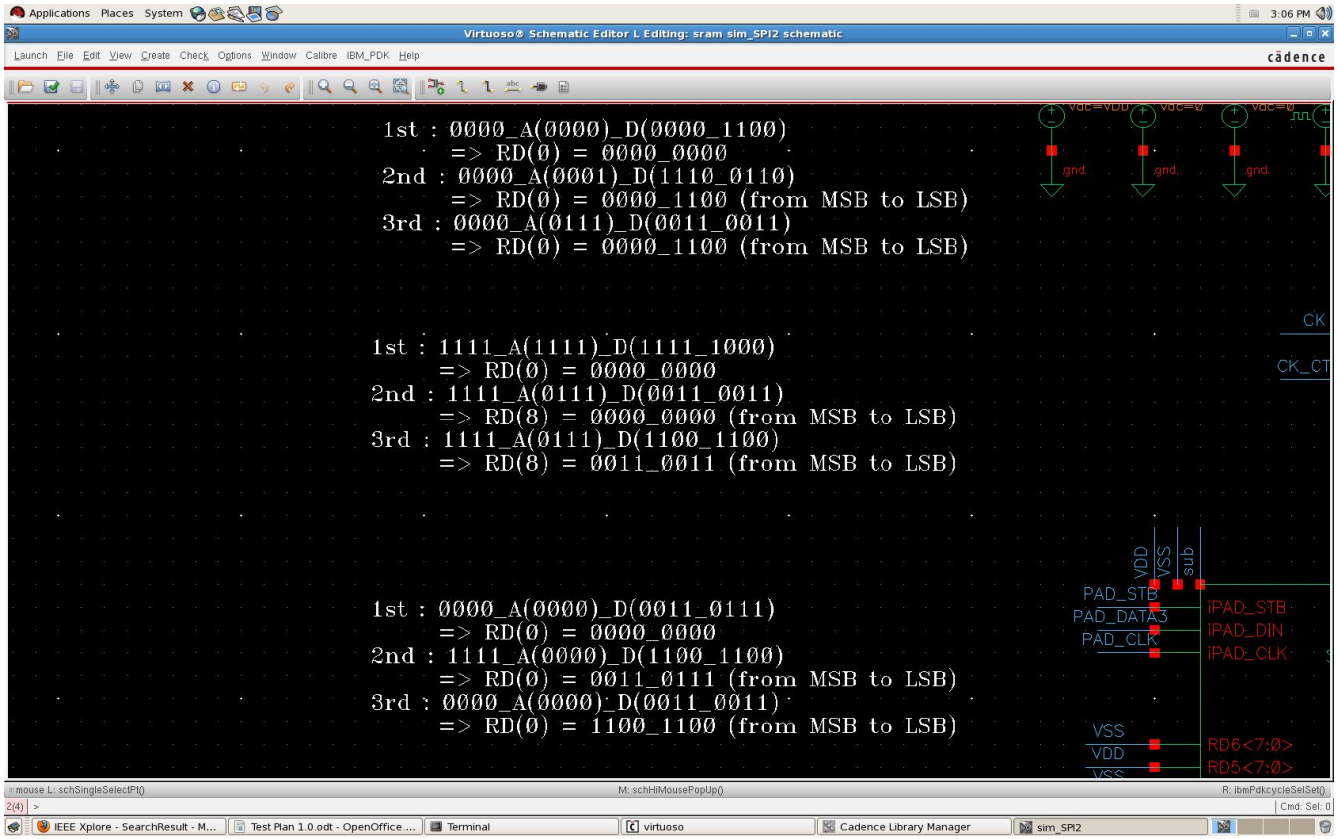
APPENDIX A

TEST SCHEME FOR THE MBC PROTOTYPE CHIP

- Make CLR high.
- Make CLR low

Repeat 32 time to write all the rows

Sample SPI I/O data



- Write SPI with the function values that need to be stored in the memory.

- Send the address and the data (for the *i-th row of the mem.*) of the write register in the SPI
- Repeat 8 times to write all the registers corresponding to the *i-th* row of the memory.

- Make CLK_EN high
- Make SCH_EN high for one CLK cycle
  - Make the SCH_EN go low after one cycle
  - Apply periodic CLR pulses
- Make write enable high.
- Write the memory.
- Make the write enable low.

- Make CLR high.
- Make CLR low
- Make CLK_EN low

Writing the schedule table and the input data

- Write SPI registers with the schedule table values.
  - Send the address and the data of the write register in the SPI
  - Repeat 8 times.

- Make CLR high.
- Make CLR low
- Adjust VDD_CLK (to set the operating frequency)
- Make CLK_EN high

Read memory and store the data in the IR and DR

- Make the VDD_Delay small to increase delay and pulse to maximum width.
- Observe IR and DR values. Increase VDD_Delay to find the optimum pulse width
- Make RD_EN and SCH_EN high.

Read the IR and DR via SPI

- Make CLK_EN low
- Send the address of the registers that need to be read in the DC register.
- Read the value of the corresponding read register in the next Clock cycle.

REFERENCES

[1] DeHon, A.; Wawrzynek, J., "Reconfigurable computing: what, why, and implications for design automation," Design Automation Conference, 1999. Proceedings. 36th , vol., no., pp.610,615, 1999

[2] V. Betz, J. Rose and A. Marquardt, “Architecture and CAD for Deep- Submicron FPGAs”, Springer, 1999

[3] J.rose and D.hill. Architectural and physical design challenges for one million gate fpgas and beyond. In FPGA, 1997

[4] K Rahmani, P Mishra and S Bhunia, “Memory-based Computing for Performance and Energy Improvement in Multicore Architectures,” GLSVLSI '12 Proceedings of the great lakes symposium on VLSI. pp 287-290, May 2012

[5] Paul, S.; Chatterjee, S.; Mukhopadhyay, S.; Bhunia, S., "Energy-Efficient Reconfigurable Computing Using a Circuit-Architecture-Software Co-Design Approach," Emerging and Selected Topics in Circuits and Systems, IEEE Journal on , vol.1, no.3, pp.369,380, Sept. 2011

[6] Paul, S.; Bhunia, S., "MBARC: A scalable memory based reconfigurable computing framework for nanoscale devices," Design Automation Conference, 2008. ASPDAC 2008. Asia and South Pacific , vol., no., pp.77,82, 21-24 March 2008

[7] Paul, S.; Bhunia, S., "Reconfigurable computing using content addressable memory for improved performance and resource usage," Design Automation Conference, 2008. DAC 2008. 45th ACM/IEEE , vol., no., pp.786,791, 8-13 June 2008